Natural Language Processing Project 3

Joel, Janet, Clement

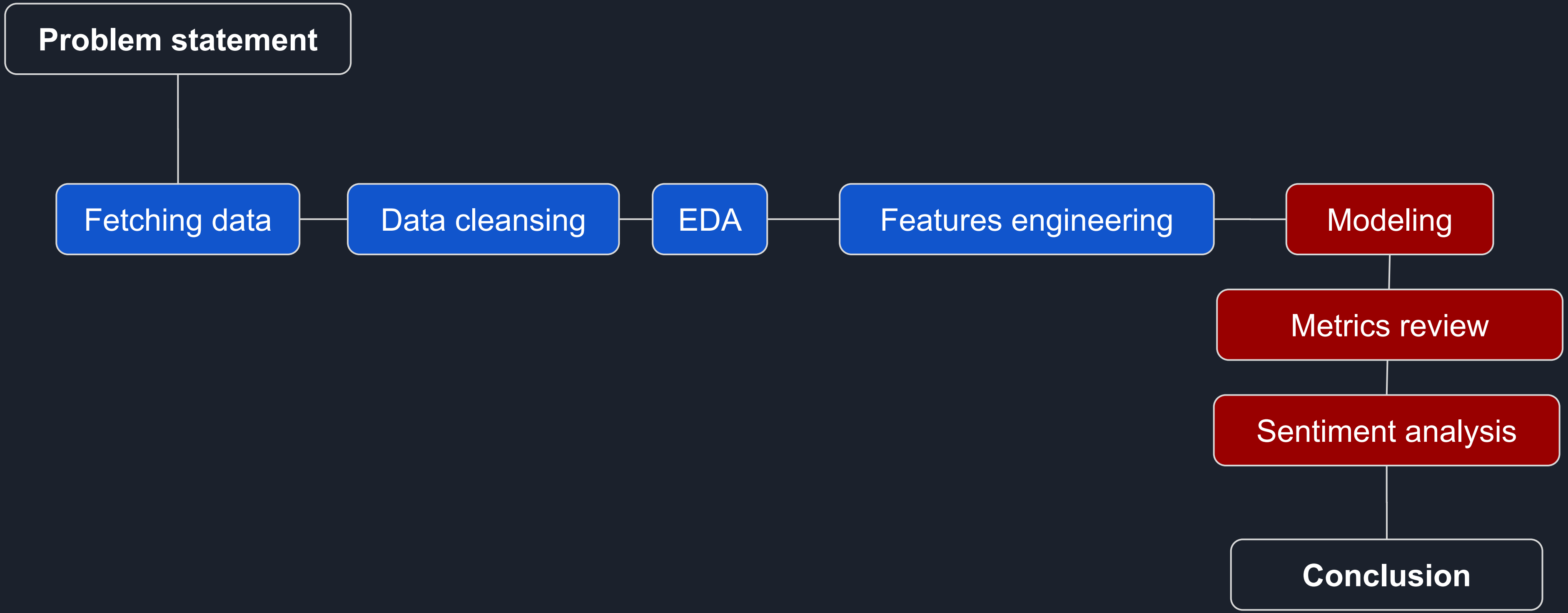
Problem statement
Can reddit to become an aggregator of conversations

Reddit now want to display organic conversations from its members, but also any publicly available conversation on the internet, and classified it under one of its existing subreddits.

They have hired us to run a "stage 0 testing" to evaluate whether such an algorithm would even be effective based sources from reddit itself, let alone importing from other sources.

DUNKIN'
DONUTS
DD
®

Plan


Problem statement

Fetching data

Data cleansing

EDA

Features engineering

Modeling

Metrics review

Sentiment analysis

Conclusion

Getting the data

Using Reddit API

Size: each subreddit got scrapped 200 post x 30 iterations = **6000 posts**

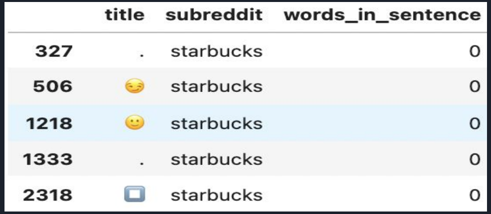
Data pre-EDA cleansing

- **Transform to lower**
- **Remove** empty posts
- **Handle** emojis

EDA

1. **Most common words** in each df
- Tokenizing
- Lemmatizing & Stemming (9498 columns vs 7381 columns)
- Removing stop words
- Checking similarity: Unigrams vs Bi-grams

2**. Counting words**
- Check for min, max and mean word count on the individual datasets
- Dropping the rows that contains no words in titles (Users include an emoji or punctuation in place of a word in title)
- Plotting in WordClouds

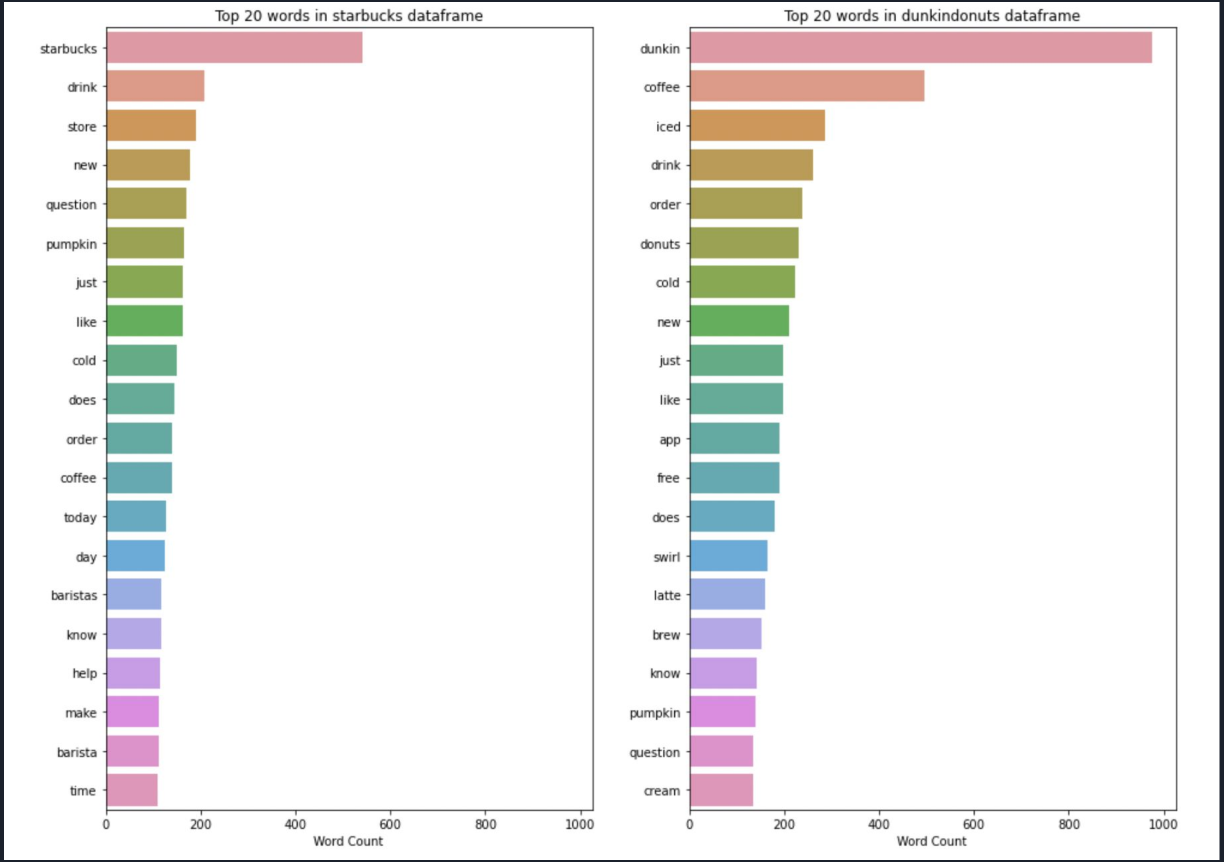


	title	subreddit	words_in_sentence
327	.	starbucks	0
506	😏	starbucks	0
1218	🙂	starbucks	0
1333	.	starbucks	0
2318	⏹	starbucks	0

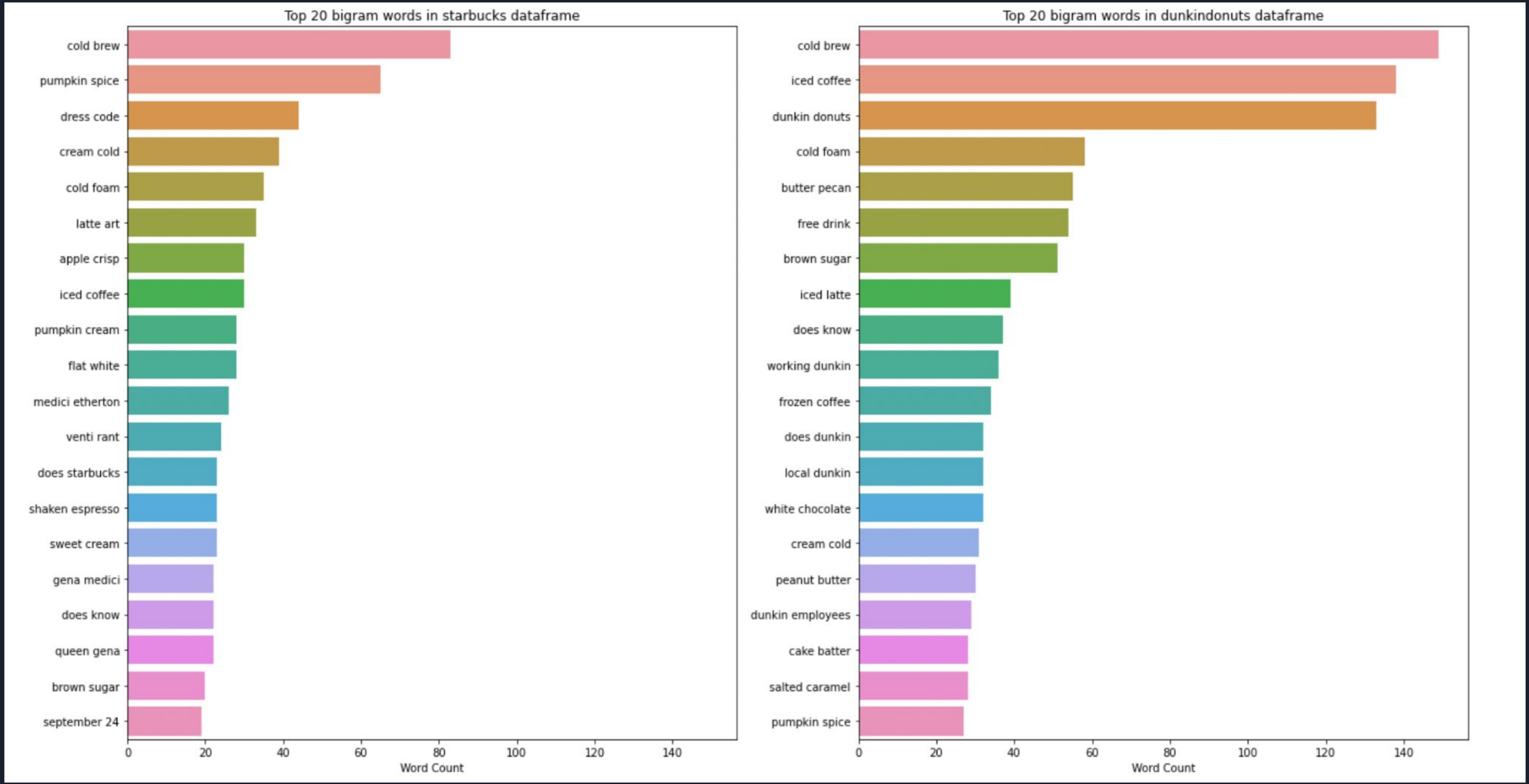
3. **Posts length** in each df

Is there a pattern in length that may influence the model?

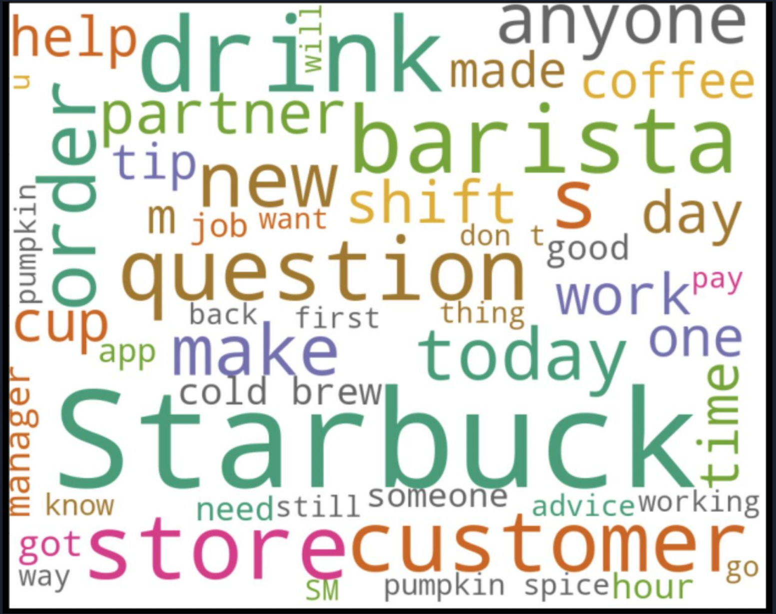
Visualization on unigram

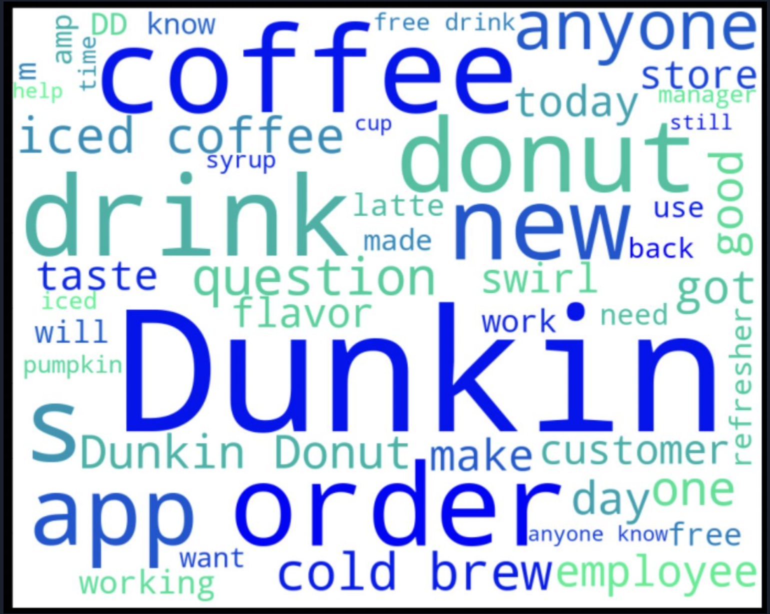


Visualization on Bigrams



WordCloud on raw title text

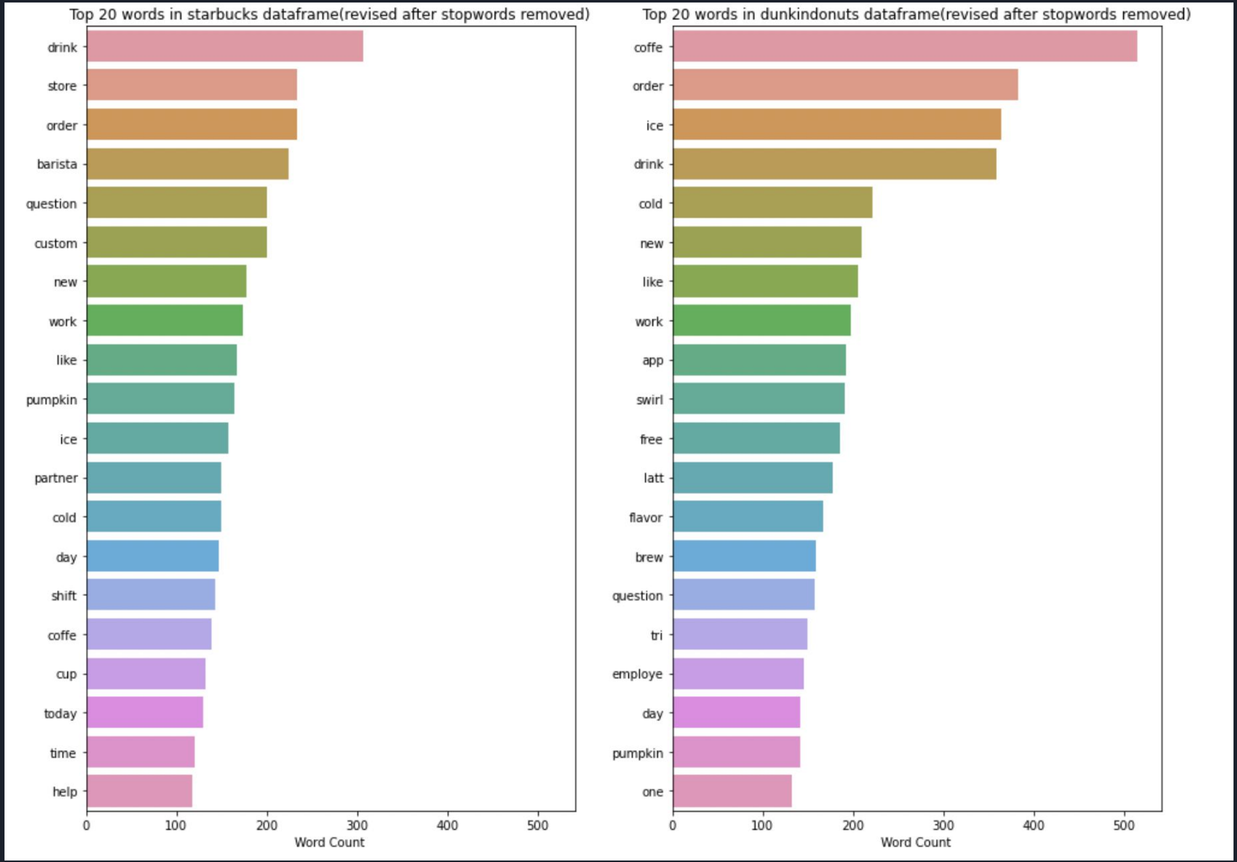




Preprocessing

- Converting text in title to lowercase
- Removing numbers
- Removing URLs
- Removing punctuation
- Applying stemming
- Remove stop words, including brand names and other miscellaneous words like ‘tall’, ‘grande’, ‘venti’ all of which describes the cup size of Starbucks

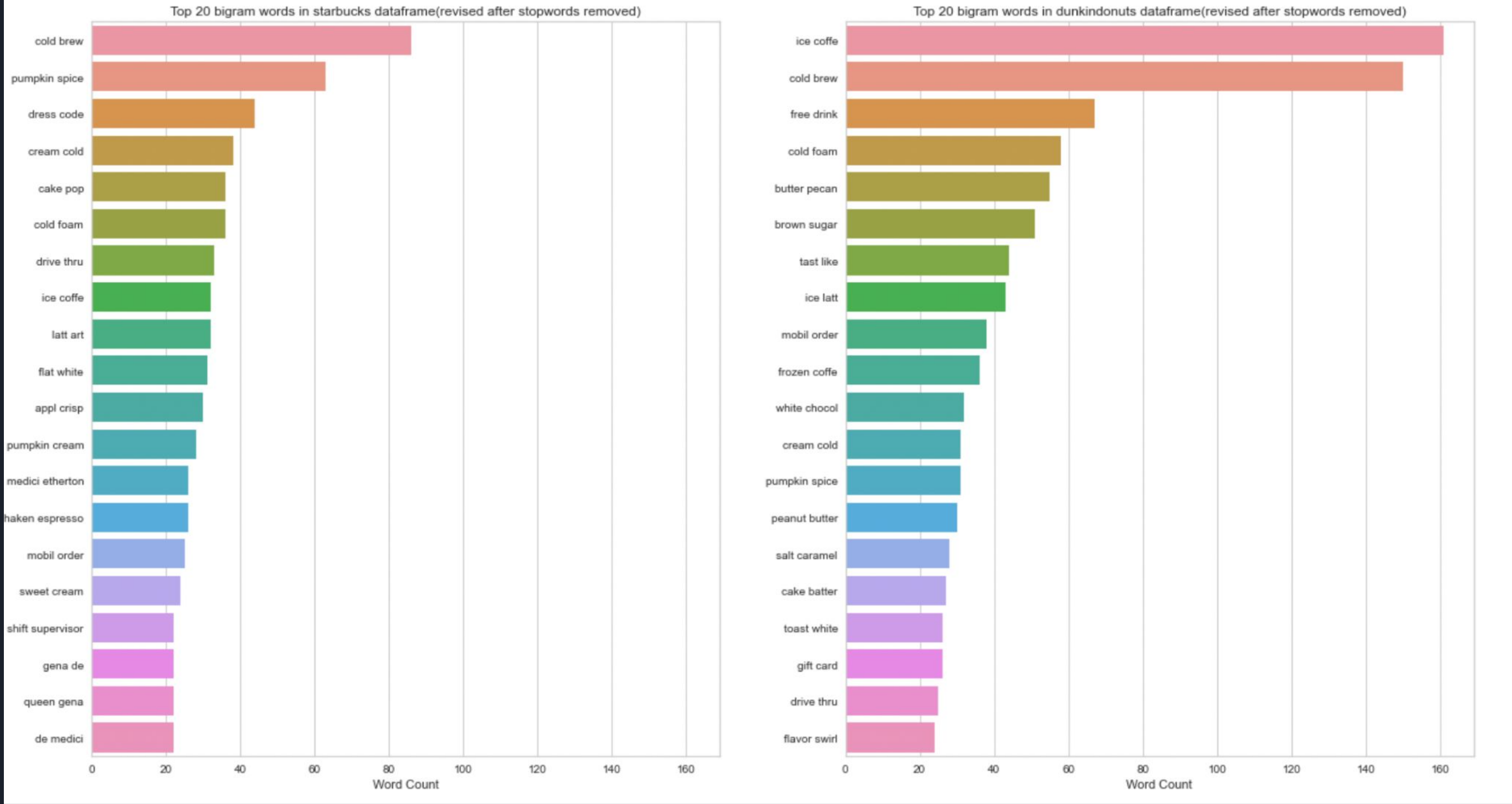
Visualization on unigram (After preprocessing)



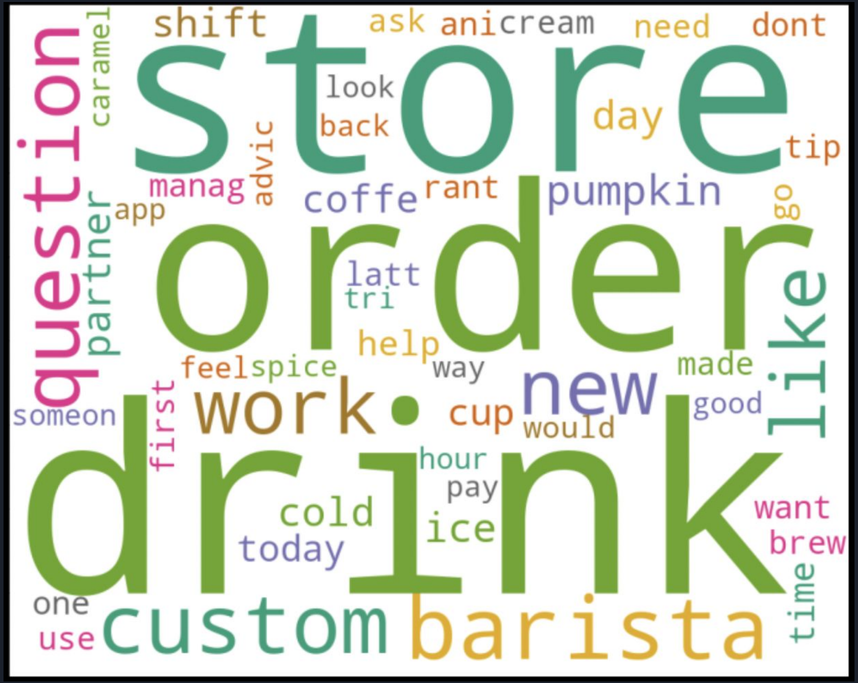
Top 20 words in starbucks dataframe(revised after stopwords removed)

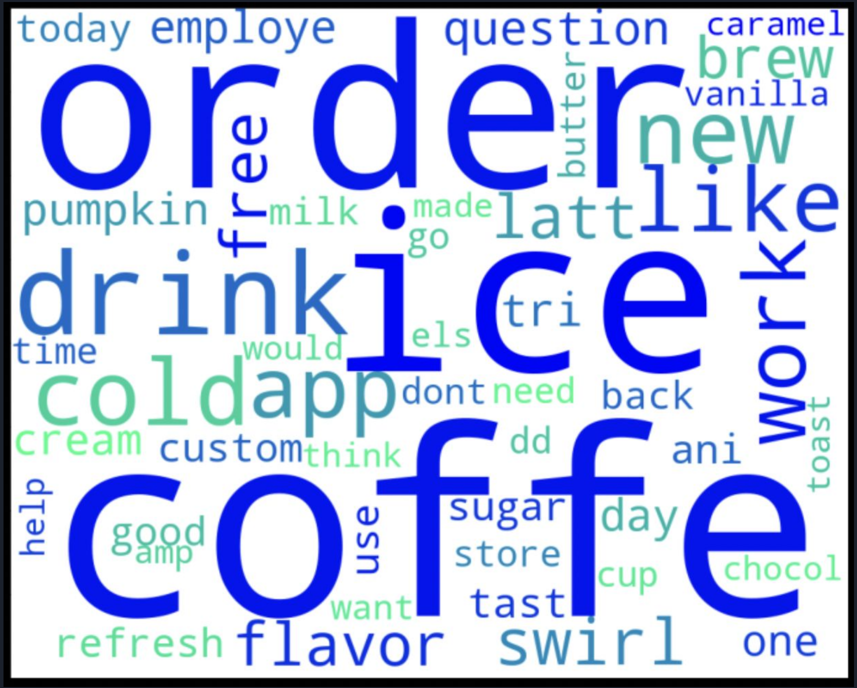
Top 20 words in dunkindonuts dataframe(revised after stopwords removed)

Visualization on Bigrams (After preprocessing)



WordClouds post-stop words removal





Classification model metrics

- Choices : Accuracy, Recall, Precision, F1

- Chosen: Accuracy
    - Target variable is balanced (0.50 each)
    - Consequences of False Positive and False Negative is more or less the same
    - No prioritizing of True Positive/ Negative or False Positive/ Negative required in this case

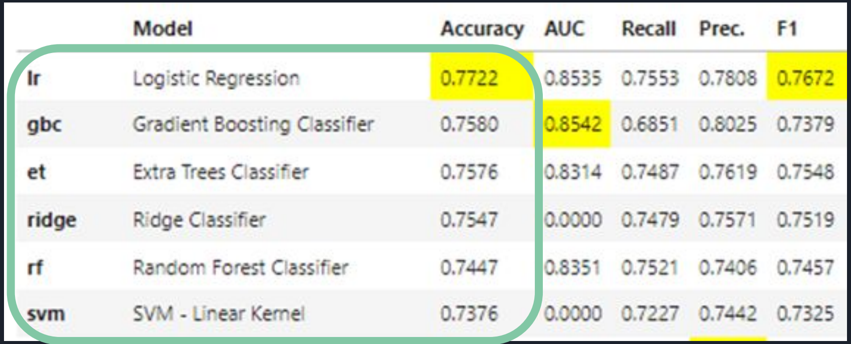
Modeling

- Baseline model = Null model = 0.50

- Compare all models in PyCaret Library

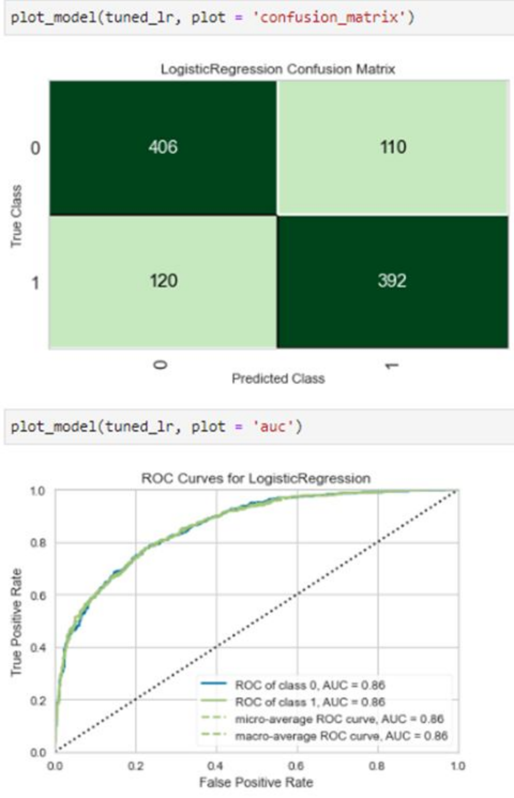
- Top models:
  - Logistic Regression
  - Gradient Boosting Classifier
  - Extra Trees Classifier

- Tune the hyperparameter



	Model	Accuracy	AUC	Recall	Prec.	F1
lr	Logistic Regression	0.7722	0.8535	0.7553	0.7808	0.7672
gbc	Gradient Boosting Classifier	0.7580	0.8542	0.6851	0.8025	0.7379
et	Extra Trees Classifier	0.7576	0.8314	0.7487	0.7619	0.7548
ridge	Ridge Classifier	0.7547	0.0000	0.7479	0.7571	0.7519
rf	Random Forest Classifier	0.7447	0.8351	0.7521	0.7406	0.7457
svm	SVM - Linear Kernel	0.7376	0.0000	0.7227	0.7442	0.7325

Models EDA

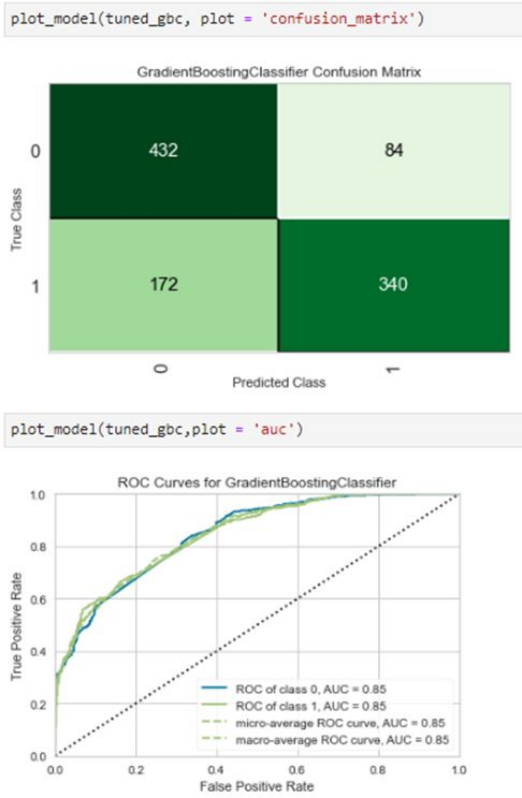


```
plot_model(tuned_lr, plot = 'confusion_matrix')
```

LogisticRegression Confusion Matrix
True Class
0
1
406
110
120
392
0
1
Predicted Class

```
plot_model(tuned_lr, plot = 'auc')
```

ROC Curves for LogisticRegression
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC of class 0, AUC = 0.86
ROC of class 1, AUC = 0.86
micro-average ROC curve, AUC = 0.86
macro-average ROC curve, AUC = 0.86

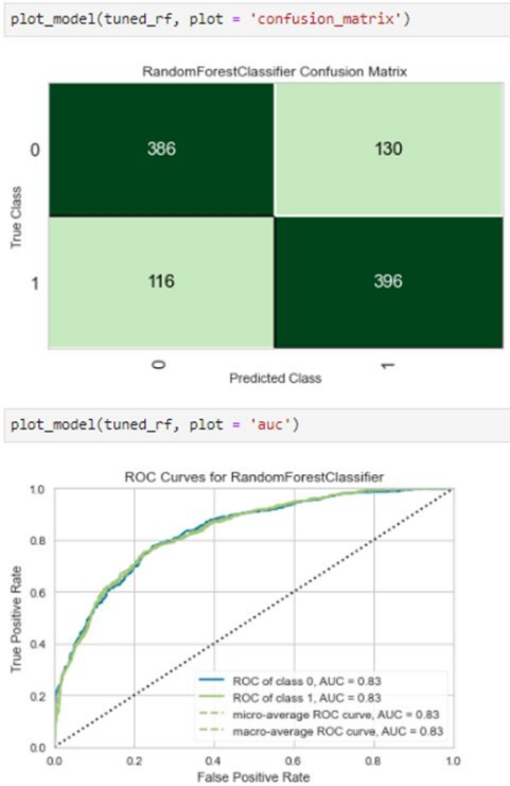


```
plot_model(tuned_gbc, plot = 'confusion_matrix')
```

GradientBoostingClassifier Confusion Matrix
True Class
0
1
432
84
172
340
0
1
Predicted Class

```
plot_model(tuned_gbc,plot = 'auc')
```

ROC Curves for GradientBoostingClassifier
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC of class 0, AUC = 0.85
ROC of class 1, AUC = 0.85
micro-average ROC curve, AUC = 0.85
macro-average ROC curve, AUC = 0.85



```
plot_model(tuned_rf, plot = 'confusion_matrix')
```

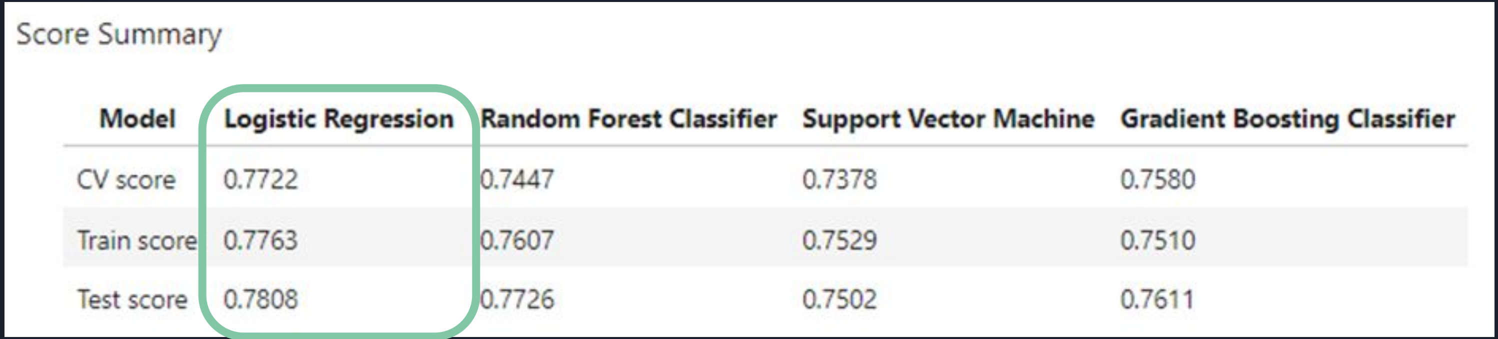
RandomForestClassifier Confusion Matrix
True Class
0
1
386
130
116
396
0
1
Predicted Class

```
plot_model(tuned_rf, plot = 'auc')
```

ROC Curves for RandomForestClassifier
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC of class 0, AUC = 0.83
ROC of class 1, AUC = 0.83
micro-average ROC curve, AUC = 0.83
macro-average ROC curve, AUC = 0.83

Final model and Evaluation

- Logistics Regression gives best accuracy of 0.78.
  - Well-fitted
  - Interpretable
  - Computational cheaper
  - Suitable for binary classification problem

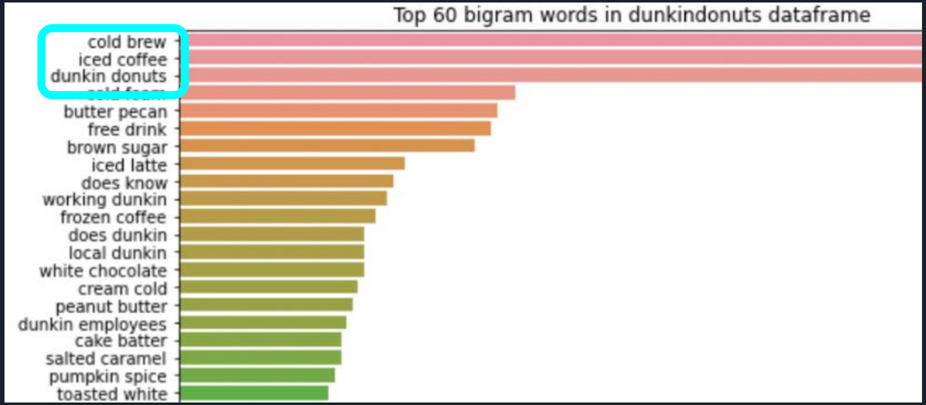


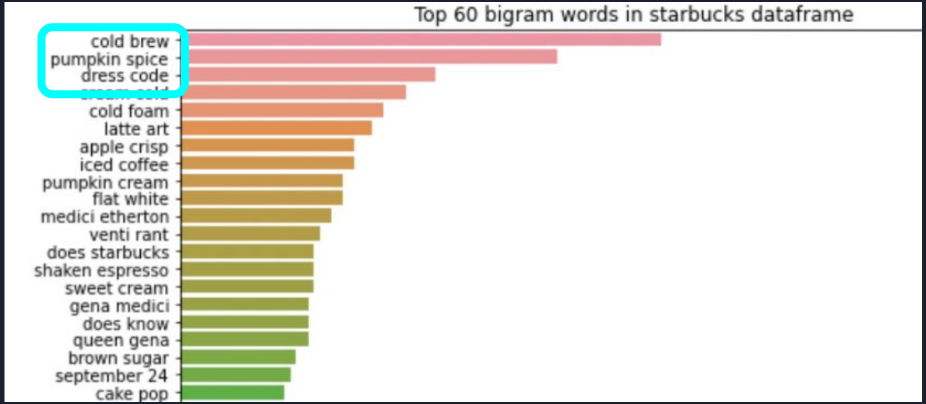
Score Summary

Model	Logistic Regression	Random Forest Classifier	Support Vector Machine	Gradient Boosting Classifier
CV score	0.7722	0.7447	0.7378	0.7580
Train score	0.7763	0.7607	0.7529	0.7510
Test score	0.7808	0.7726	0.7502	0.7611

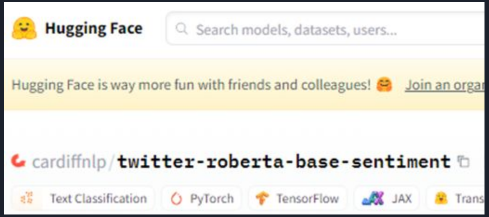
Sentiment Analysis - on Hot Topics

- Dunkin':
  - Cold brew
  - Iced coffee
  - Dunkin donuts
- Starbucks:
  - Cold brew
  - Pumpkin spice
  - Dress code
- Both:
  - Reward
  - Service





Sentiments


Hugging Face

Search models, datasets, users...

Hugging Face is way more fun with friends and colleagues! 🤗 Join an orga

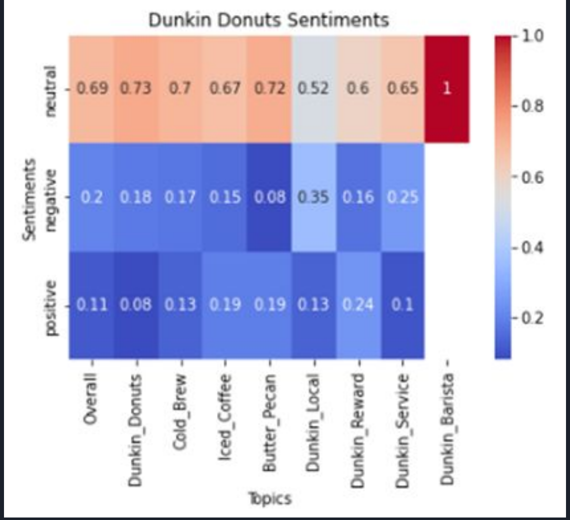
cardiffnlp/twitter-roberta-base-sentiment

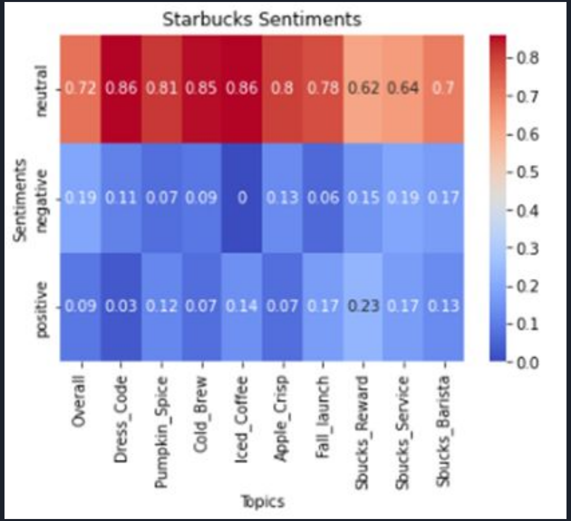
Text Classification · PyTorch · TensorFlow · JAX · Trans

Majority posts seem neutral - Questions, Discussions

Well-received topics : Rewards (free drinks, free beverages, points)

Negative topics: Services (mobile order, app, staffs)





Project conclusion

We have obtained a **reliable classifier to differentiate between 2 posts on reddit;**

__**Recommendation:**__
We can use Logistic Regression as our model to allocate subreddits with satisfying accuracy.

We can add on Sentiment Analysis feature to ease the navigation inside a subreddit for **HR | marketing business development teams**

__**However**__
To accurately classify a post coming from anywhere on the internet **will prove way more difficult**

Future works

Gathering more data - fine tune model and enable time-series analysis

Text-pre-processing steps

Improving on our models training (Gridsearch/ vectorizers etc...)

Stacking models to make predictions that have better performance than any single model in the ensemble